Software

Support for Diagnosis of Custom Computer Hardware

The Coldfire SDN Diagnostics software is a flexible means of exercising, testing, and debugging custom computer hardware. The software is a set of routines that, collectively, serve as a common software interface through which one can gain access to various parts of the hardware under test and/or cause the hardware to perform various functions. The routines can be used to construct tests to exercise, and verify the operation of, various processors and hardware interfaces. More specifically, the software can be used to gain access to memory, to execute timer delays, to configure interrupts, and configure processor cache, floating-point, and direct-memory-access units.

The software is designed to be used on diverse NASA projects, and can be customized for use with different processors and interfaces. The routines are supported, regardless of the architecture of a processor that one seeks to diagnose. The present version of the software is configured for Coldfire processors on the Subsystem Data Node processor boards of the Solar Dynamics Observatory. There is also support for the software with respect to Mongoose V, RAD750, and PPC405 processors or their equivalents.

This program was written by Dwaine S. Molock of Goddard Space Flight Center. For further information, contact the Goddard Innovative Partnerships Office at (301) 286-5810. GSC-15478-1

Providing Goal-Based Autonomy for Commanding a Spacecraft

A computer program for use aboard a scientific-exploration spacecraft autonomously selects among goals specified in high-level requests and generates corresponding sequences of low-level commands, understandable by spacecraft systems. (As used here, "goals" signifies specific scientific observations.) From a dynamic, onboard set of goals that could oversubscribe spacecraft resources, the program selects a non-oversubscribing subset that maximizes a quality metric. In an early version of the program, the requested goals are assumed to have fixed starting times and durations. Goals can conflict by exceeding a limit on either the number of separate goals or the number of overlapping goals making demands on the same resource.

The quality metric used in this version is chosen to ensure that a goal will never be replaced by another having lower priority. At any time, goals can be added or removed, or their priorities can be changed, and the "best" goal will be selected. Once a goal has been selected, the program implements a robust, flexible approach to generation of low-level commands: Rather than generate rigid sequences with fixed starting times, the program specifies flexible sequences that can be altered to accommodate run time variations.

This program was written by Gregg Rabideau, Steve Chien, and Ning Liu of Caltech for NASA's Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1).

This software is available for commercial licensing. Please contact Karina Edmonds of the California Institute of Technology at (626) 395-2322. Refer to NPO-44541.

Dynamic Method for Identifying Collected Sample Mass

G-Sample is designed for sample collection missions to identify the presence and quantity of sample material gathered by spacecraft equipped with end effectors. The software method uses a maximum-likelihood estimator to identify the collected sample's mass based on onboard force-sensor measurements, thruster firings, and a dynamics model of the spacecraft. This makes sample mass identification a computation rather than a process requiring additional hardware.

Simulation examples of G-Sample are provided for spacecraft model configurations with a sample collection device mounted on the end of an extended boom. In the absence of thrust knowledge errors, the results indicate that G-Sample can identify the amount of collected sample mass to within 10 grams (with 95-percent confidence) by using a force sensor with a noise and quantization floor of 50 micrometers. These results hold even in the presence of realistic parametric uncertainty in actual spacecraft inertia, center-of-mass offset, and first flexibility modes.

Thrust profile knowledge is shown to be a dominant sensitivity for G-Sample, entering in a nearly one-to-one relationship with the final mass estimation error. This means thrust profiles should be well characterized with onboard accelerometers prior to sample collection. An overall sample-mass estimation error budget has been developed to approximate the effect of model uncertainty, sensor noise, data rate, and thrust profile error on the expected estimate of collected sample mass.

This program was written by John Carson of Caltech for NASA's Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1).

This software is available for commercial licensing. Please contact Karina Edmonds of the California Institute of Technology at (626) 395-2322. Refer to NPO-44403.

Optimal Planning and Problem-Solving

CTAEMS MDP Optimal Planner is a problem-solving software designed to command a single spacecraft/rover, or a team of spacecraft/rovers, to perform the best action possible at all times according to an abstract model of the spacecraft/rover and its environment. It also may be useful in solving logistical problems encountered in commercial applications such as shipping and manufacturing.

The planner reasons around uncertainty according to specified probabilities of outcomes using a plan hierarchy to avoid exploring certain kinds of suboptimal actions. Also, planned actions are calculated as the state-action space is expanded, rather than afterward, to reduce by an order of magnitude the processing time and memory used. The software solves planning problems with actions that can execute concurrently, that have uncertain duration and quality, and that have functional dependencies on others that affect quality. These problems are modeled in a hierarchical planning language called C_TAEMS, a derivative of the TAEMS language for